International Journal of Research in Advent Technology (E-ISSN: 2321-9637) Special Issue
1st International Conference on Advent Trends in Engineering, Science and Technology
"ICATEST 2015", 08 March 2015

Timetable Scheduling by Genetic Algorithm

Girish Umaratkar[1], Sachin Murab[2]
[1] *M.E. Student JCET Yavatmal(MS),India girishumaratkart@gmail.com*
[2] *Assistant Professor,JCET,Yavatmal(MS),India, sachinmurab21@gmail.com*

Abstract: The problem is to implement an algorithm to create a semester course timetable by assigning time-slots and rooms to a given set of course to be run in that semester in given constraints. the constraints includes avoiding clashes in time slots and room, assigning appropriate rooms and appropriate number of slots and contact hours to the course etc. Although most of the college administrative work has been computerized, the lecture time table scheduling is still mostly done manually due to its technical difficulties[1]. The manual scheduling lecture-timetable require considerable time and efforts. The lecture-timetable scheduling is constraint satisfaction problems in which we find an optimal solution that satisfy the given constraints. The college lecture-timetabling problem ask us to find some time slots and classrooms which satisfy the constraints imposed on offered course, instructors, classroom and so on. Therefore , the variable to be instantiated are time slots and classrooms of offered courses. Since this problem is a combinational optimization problem belonging to NP-Hard class, the computation time for timetabling tends to grow exponentially as the number of variable increase[2-3]. There have been a number of approaches made in past decades to the problem of constructing timetables for collages and schools.

Keywords: timetable scheduling problem, genetic algorithm, 3D representation

1. INTRODUCTION

Scheduling a timetable could also be represented like special class of 3D cutting problems[5]. The timetable could be presented as a 3D structure. The dimensions of 3D timetables are: days (x-axis), timeslots (y-axis) and rooms (z-axis). The classes are shown as cubes, which should be placed in a 3D timetable structure (Fig.

The scheduling is a process of placing those cubes into a timetable , in the way that no conflicting classes (which allocate the same resource, a student group or an instructor) are placed in the same timeslot. The timetable scheduling process could be formally defined with binary variables xcdtrgi, which have the value of 1 if and only if instructor i lectures the class c on day d at time t, for group g in room r.

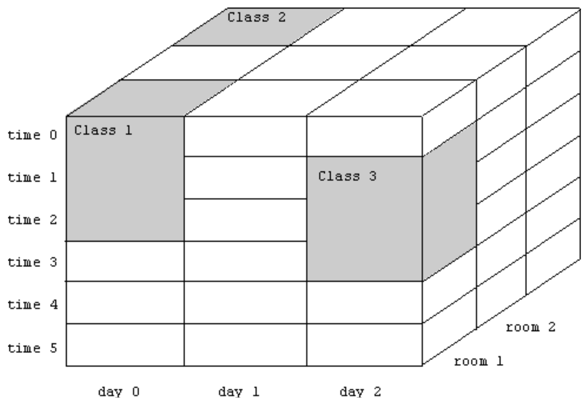

Fig 1. 3D representation of class

2. CONSTRAINT

2.1Hard Constrain

Hard constraints are rigidly enforced. Examples of such constraints are: No lecturer should have different classes at the same time slot.

There cannot be more than 2 classes for a subject on one day, For each time period there should be sufficient resources (e.g. rooms and lecturers) available for all the events that have been scheduled for that time period.

2.2 Soft Constrain:

Soft constraints are those that are desirable but not absolutely essential. In real-world situations it is, of course, usually impossible to satisfy all soft constraints. Examples of soft constraints (in both exam and course timetabling) are: Every staff should get at least one first hour

_ Lecturer having two theory subjects has no lab assignments

_ Lecturer having one theory may get two lab classes.

_ A particular class may need to be scheduled in a particular time period.

_ Lab Classes may not be in consecutive hours.

_ Lecturers may prefer to have all their lectures in a number of days and to have a number of lecture-free days.

3. THE CLASS TIMETABLE PROBLEM

Time Table problems are mainly classified as constraint satisfaction problems, where the main goal is to satisfy all problem constraints, rather than optimizing a number of objectives. At present, science has no analytical solution method for all problem cases of this category, other than exhaustive search, which however cannot be applied but only to toy problems. Due to the immense search spaces, Automated timetable scheduling, on the other hand, is a task of great importance as it can save a lot of man-hours work, to institutions and companies, and provides optimal solutions with constraint satisfaction.Scheduling is the arrangement of entities (people,tasks, vehicles, lectures, exams, meetings, etc.) into a pattern in space-time in such a way that constraints are satisfied and certain goals are achieved[5] . Constructing a schedule is the problem in which time, space and other (often limited) resources have to be considered in the arrangement.Holland's original schema was a method of classifying objects, then selectively"breeding"those objects with each other to produce newobjects to be classified . Created for the direct purpose of modeling Darwinian natural selection, the programs followed a simple pattern of the birth,mating and death of life forms.

A top-level description of this process is given in figure

```
Create a population of creatures.
Evaluate the fitness of each creature.
While the population is not fit enough:
{
Kill all relatively unfit creatures.
While population size< max;
{
Select two population members.
Combine their genetic material to create a new creature.
Cause a few random mutations on the new creature.
Evaluate the new creature and place it in the population.
}
}
```

Genetic algorithms (GAs) are evolutionary algorithms that use the principle of natural selectionto evolve a set of solutions toward an optimum solution. GAs are not only very powerful, but are also very easy to use as most of the work can be encapsulated into a single component, requiring users only to define a fitness function that is used to determine how "good" a particular solution is relative to other solutions.The creatures upon which the genetic algorithm acts are composed of a series of units of information- referred to as genes. The genes, which make up each creature, are known as the chromosome. Each creature has its own chromosome.A GA, as shown in figure 1 requires a processof initializing, breeding, mutating, choosing and killing. The order and method of performing each of these gives rise to many variations on Holland's original schema.

4. THE GENETIC ALGORITHM IMPLEMENTATION

Genetic algorithms are adaptive systems inspired by natural evolution[3]. They can be used as techniques for solving complex problems an for searching of large problem spaces[6]. Genetic algorithms are belonging to guided random search techniques, which try to find the global optimum. J.H. Holland presented this concept in early seventies. The power of genetic algorithms and other similar techniques (simulated annealing, evolutionary strategies) lies in the fact that they are capable to find global optimum in multi-modal spaces (spaces with many local optimums). Classical gradient methods will always gravitate from starting position to some local optimum, which could also be global, but it can not be determined for certain. Genetic algorithms are working with the set of potential solutions, which is called *population*. Each solution item (*individual*) is measured by *fitness function*.[4] The fitness value represents the quality select individuals with better genetic material for producing new individuals and further generation

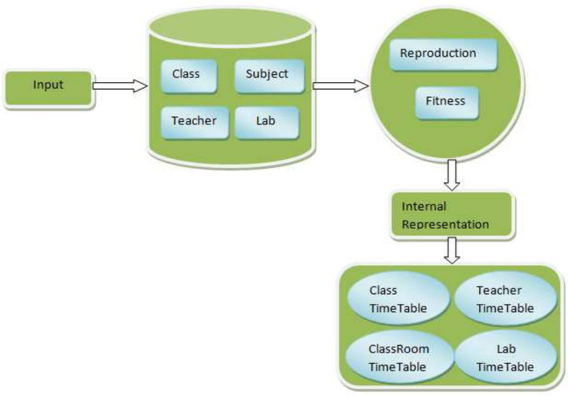

Fig 2. Architectural Diagram

4.1 TimeTable Scheduling Using Genetic Algorithm

FitnessFunction()

Begin
For teacher=1 to N do
Check constraint1() {for each teacher it checks that the lectures come according to predefined load in one row}
Check constraint2() {for each teacher it checks that the lectures are arranged according to the division i.e. lecture is in morning session or afternoon session}
Check constraint3() {for each teacher it checks the lab constraints i.e.Clashes in batches and subjects for the lab}
Check constraint4(){for each teacher it checks the column constraint i.e. for a given time slot one teacher can take onlyone lecture that is one teacher cannot take the lecture on different class in different class rooms at a time}
end for;
call to Genetic Operator Reproduction;
End;

ReproductionOp()
Begin
For teacher=1 to N do
If (no of lectures < load) then
add that lecture in unused location.
End If;
If (lectures are not arranged according to division)Then adjust the lectures in their respective division in unused location.
End If;
If (lab constraints are not satisfied) then
adjust the labs according to availability.
End If;
If (column constraints are not satisfied) then
adjust the lecture or practical in the next column starting from the first column until the column constraints are satisfied.
End If;
End for;
End;

10005@0010004@5@10104@6@00006@000000000000000000
9#00000000000207#0020000000000007#000028#000000
20006#5#20005#0020004#0020004#6#0000000000000000
02335#0002004#6#3200000030000000000000000000000
33009$8$33004$7$00006$5$000000000000000000000000
0022000001130000013200000330000000000000000000000
009#00029#8#00028#0000027#0020007#00000000000000
7#002000007#2000008#02009#0002009#000000008#0000
00006$9$00308$4$0303000000003007$0000000000005$00
00005$000000009$0000008$00006$4$3300000033007$00
00000000000206#0000205#6#0200000020005#4#00004#00
8@9@00008@0000007@0010017@9@0000000000000000000
01104@5@00005@6@00016@0010000000000004@000000000
00000220000000200000000100000000200000101000000000
7@0010100000010019@7@00008@000000009@0000008@0000
00010000000010000000000000011200000220000000000000
0000010000000100008@00000000010027@0022009@000000
0000004@00000000000005@000015@0011006@000000000

Fig 3 Internal Representation

4.2 GenerateTimeTable()

Begin
If(fitness == (no of teachers * no of constraints)) then
Generate Teacher TimeTable in printable format;
Generate Class Room TimeTable in printable format;
Generate Lab TimeTable in printable format;
Generate Class TimeTable in printable format;
End If;
End;

4.3 Genetic Algorithm

1.Generate a random population of chromosomes
2.If the termination criteria are satisfied, stop. Else, continue with step 3
3.Determine the fitness of each chromosome
4.Apply crossover and mutation to selected chromosomes from the current generation, to generate a new population of chromosomes – the next generation
5.Return to step 2

Algorithm
Initialization
{ this routine creates a population of N individuals with input characters- 0,1,2,3,4,5,6,7,8,9,@,#,$ }
Begin
For i=1 to N do

```
        evolve population.
        Check for fitness.
If(fitness) then break and generate
  TimeTable
 {
  fitness = no of teachers * no of constraints
  }
  Else
                continue evolve population.
        End If;
End For;
End;
```

5. CONCLUSION

The initial scheduling problem with large number of binary variables has been reduced to the acceptable size by eliminating certain dimensions of the problem and encorporating those dimensions into constraints. The grouping of several binary variables into one gene value significantly reduced the individual size. Now it is possible to try to solve the full size problem (problem of the whole FER schedule) with genetic algorithm approach. Such a representation of the scheduling problem achieves the acceptable algorithm speed, so small size problems are solved in tensseconds. Significant improvements have been achieved by using intelligent operators. The intelligent algorithm converges much faster then the basic algorithm and represents a good starting point for complete solving of the full scale problem.

Future Enhancement

1. College Unit Test TimeTable
2. Messaging to Teacher before lecture

REFERENCES

[1]. Ehrgott M., Gandibleux X.(2000), A Survey and Annotated Bibliography of Multiobjective Combinatorial Optimization, OR Spectrum, Vol. 22, No. 4, Springer, pp. 425-460.

[2].M. Golub, D. Jakobovic, "Genetic algorithm", parts I and II (in Croatian), Faculty of electrical engineering and computing, Zagreb, 1997/2002

[3] M. Golub, S.Kasapovic, "Scheduling Multiprocessor Tasks with Genetic Algorithms", Proceedings of the IASTED International Conference Applied Informatics, Insbruck, Austria, pp. 273-278, February 18, 2002.,

[4]Sait S.Ali.,and Benten M.:"Scheduling and allocation in high level synthesi using stochastic technique",Microelectron.J.,1996,27,pp.693-712.
[5]International timetabling competition accessed march-2005.http://www.idsia.ch/Files/ttcomp2002/

[6]M. Golub, S.Kasapovic, "Scheduling Multiprocessor Tasks with Genetic Algorithms", Proceedings of the IASTED International Conference Applied Informatics, February 18-21, 2002., Insbruck, Austria, pp. 273-278, available via ftp from: http://www.zemris.fer.hr/~golub/clanci/ai2002/ai2002.doc, February 18-21, 2002.,